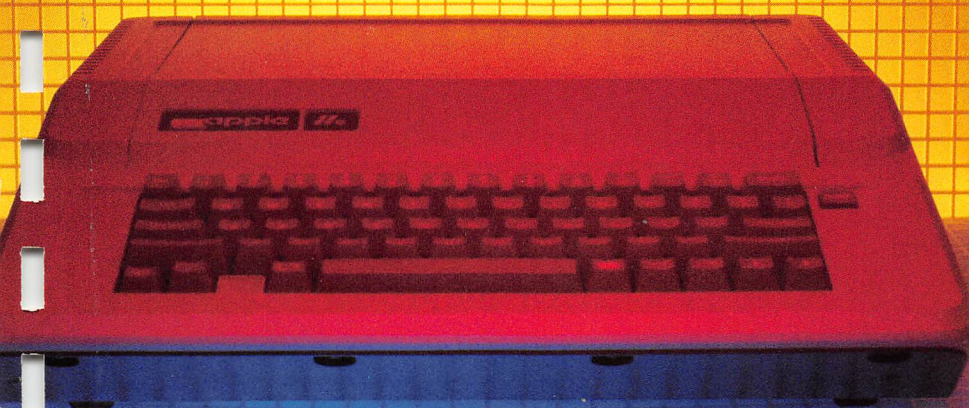
1603

COMPUTER COMPANION FOR THE APPLE II®/APPLE *IIe*®



BY ROBERT P. HAVILAND

COMPUTER COMPANION
FOR THE
APPLE II®/APPLE *IIe*®

BY ROBERT P. HAVILAND

TAB TAB BOOKS Inc.
BLUE RIDGE SUMMIT, PA. 17214

FIRST EDITION

FIRST PRINTING

Printed in the United States of America

Library of Congress Cataloging in Publication Data

Haviland, Robert P.
Computer companion for the Apple II/Apple IIe.

Includes index.
1. Apple II (Computer) I. Title.
AQ76.8.A662H29 1983 001.64 83-4964
ISBN 0-8306-1603-9 (pbk.)

Contents

Introduction

This little manual is intended to be a constant companion of the Apple II® family of personal/business computers, including the Apple II +, the FRANKLIN ACE 1000™, and the "look-a-like" ORANGE™, and PINEAPPLE™. It is intended for use in creating programs and getting them running.

Each index tab opens to a key word. The book includes all key words used by APPLE MICROSOFT BASIC™. For each key word, you are given:

The *Name*, if different than the key word
The *Token* used for internal storage
The *Class* of instruction
The required use *Form*
The *Conditions* under which the key word should be used and the result of its use

In addition, some entries give warnings and hints on the use of the key word.

Appendices provide summary information on variables, delimiters, operators, and priorities plus various other data necessary or useful in programming, including DOS commands and machine-language operations codes.

In preparing this material, the ultimate authority has been the Apple II + computer itself, in 48K form with disk. The manufacturer's literature is the next level of authority, and general literature the last. All stated forms have been checked and are correct at the time of writing.

The author, editor, and publisher would appreciate a note if you find any errors that may have crept in, indications of changes by the manufacturer, or suggestions for making the volume more useful.

Token (none) **Name** Ampersand

Class System and Utility

&

Conditions Intended for internal use; not a proper command.

If used, causes an unconditional jump to location $3F5, then to $FF58.

Use RESET to escape.

Prints register contents.

Token 212 **Name** Absolute Value

Class Mathematical Function

ABS

Forms ABS (variable)

ABS (expression)

Conditions Applied to a variable or an expression, returns the absolute (positive only) value of the variable or expression.

ABS has no meaning for string variables or strings.

Token 205

Class Arithmetic Relational Operator

AND

Form Relation 1 AND relation 2

Conditions The logical operator AND assumes the value true only if both relation 1 and relation 2 are true.

May be combined with NOT, to give the AND NOT relation, or NAND.

If both stated relations include NOT, the result is equivalent to OR.

There may be multiple ANDs in an expression.

Has no meaning for string variables.

A relation is true if nonzero.

Truth table for AND

X	Y	X AND Y
F	F	F
F	T	F
T	F	F
T	T	T

Token 230 **Name** Code

Class Array and String Function

ASC

Forms ASC (string variable)

ASC (string expression)

ASC ("string")

Conditions Returns the ASCII code of the first character of the variable, expression, or string.

The string must be in quotes and must not be empty nor include quotes.

The returned value may not be the lowest number for a given character.

Has no meaning for numerics.

Token 225 **Name** Arctangent

Class Mathematical Function

ATN

Forms ATN (number)

ATN (variable)

ATN (expression)

Conditions Returns the angle, in radians, whose tangent is equal to the specified value or the current value of a variable or expression; integer values are converted to real values before evaluation.

Returned values are in range $-\pi/2$ to $+\pi/2$ radians.

ATN has no meaning for string quantities.

Token 197

Class Graphic and Game

AT

Forms Low Resolution Graphics

HLIN expression 1, expression 2
AT expression 3

VLIN expression 1, expression 2
AT expression 3

High Resolution Graphics

DRAW expression 1 AT expression 2, expression 3

XDRAW expression 1 [AT expression 2, expression 3]

Conditions In low resolution graphics (GR), AT is used to specify the row (HLIN) or column (VLIN) for which a line is to be drawn.

In high resolution graphics (HGR), AT is used to specify the X, Y coordinates at which a shape (expression 1) is to be drawn.

See HLIN, VLIN, DRAW, GR, HGR.

The terms in brackets are optional. (See XDRAW).

Token 140

Class System and Utility Command

CALL

Form CALL address

Conditions Used to call a machine language routine starting at indicated address.

Address may be positive or negative and range from −65535 through 65535. It may be a variable or an expression.

(CALL −936 ≡ CALL 64600)

Rules of 6502 machine language apply to called routines.

A called routine should end with an RTS command.

Note CALL −151 transfers control to the monitor. See Appendix K for other useful calls.

Token 231 **Name** Character

Class Array and String Function

CHR$

Forms CHR$ (number)

CHR$ (variable)

CHR$ (expression)

Conditions Returns the ASCII character whose code is the value of the expression.

Value must be in range 0 to 255 inclusive.

Real values are converted to integers.

CHR$ has no meaning for string variables.

Token 189

Class Editing and Format Command

CLEAR

Conditions Sets values of all variables and arrays to zero, and all string variables and arrays to the null value.

Resets pointers and stacks used by the operating system.

Does not delete the program.

Note In multiple runs with one or more changing variables, reinitialization of parameters will be required after CLEAR.

Token 160

Class Graphic and Game Command

COLOR=

Forms COLOR = number

COLOR = expression

Conditions Sets the color for plotting in the low resolution mode.

Numbers and expressions are evaluated modulo 16, and are limited to the range 0 to 255 inclusive. If real, they are converted to integer form.

COLOR is set to 0 by the GR command.

In the text mode, COLOR affects PLOT characters.

In high resolution graphics, COLOR is ignored.

See Appendix H, Codes and Designators for COLOR.

Token 187 **Name** Continue

Class System and Utility Command

CONT

Conditions Used to resume program execution after a STOP, END, or control C command. Execution resumes at next instruction, *not* next line number.

May not operate properly after control C, since this clears some pointers and stacks, for example, during INPUT execution.

Results in error message after a halt if

—any program line is deleted or modified.

—any error message has occurred.

(However, variables may be modified during the halt.)

If CONT is used within a program it causes a halt, with blocked program control. Use control C to regain control.

Token 222 **Name** Cosine

Class Mathematical Function

COS

Forms COS (number)

COS (variable)

COS (expression)

Conditions Returns the cosine of an angle expressed in radians.

COS has no meaning for string quantities.

Token 131

Class INPUT OUTPUT Command

DATA

Form DATA Item 1, Item 2, ...

Conditions Creates a list of items which can be used by READ statements.

Items can be literals, strings (enclosed in quotes), or numerical values, in order, separated by commas.

Items in DATA statements must be added in the order in which the line numbers are encountered and then in the order in which the items are to be used.

DATA can appear anywhere in a program, even before the READ command.

A numeric item cannot include a comma.

In a string, spaces before first element and following the string are ignored.

A Quotation Mark with a DATA item

causes an error message, but all other characters are accepted except Control X and Control M.

In a literal, the colon, comma, Control X and Control M are not accepted. An initial quotation mark is not accepted.

A nonexistent element occurs if

- —There is no nonspace character between two commas.
- —A comma is the last nonspace character before RETURN.

If READ requires an input for a nonexistent element, it is interpreted as a zero if a numeric is required, or as a null if a string is required.

See INPUT, READ.

Token 184 **Name** DEFine (Function)

Class Input-Output Command

DEF

Form DEF FN name (argument) = expression

Conditions Used with FN to define a function for subsequent use.

The expression may be only one statement in length.

The named function may be redefined at any point in the program, with the current definition being used when named function is again encountered.

The expression may contain any number of variables, but integer variables are not allowed.

The argument following name is a dummy variable, replaced at FN use.

The variable following the name does not need to appear in the defining expression. In this case the using FN argument is ignored.

The defined function is used by the Form: FN name (argument), where argument may be a number, a variable, or an expression.

String functions may not be defined. The name and variable may not be integer types.

DEF FN cannot be used in the immediate execution mode.

See FN.

Note Only the first two characters in a variable name are evaluated. The rest are ignored unless they include a reserved word, which produces an error input.

Token 133 **Name** Delete

Class Editing and Format Command

DEL

Form DEL line number 1, line number 2.

Conditions Deletes the line or lines from line number 1 to line number 2, inclusive.

Is ignored if line number 1 is greater than line number 2.

If line number 1 does not exist in the program, deletes the next larger line number.

If line number 2 does not exist in the program, deletes the next smaller line number.

If encountered in program execution, deletes stated lines, then halts execution, which cannot be resumed by CONT.

Note For single line deletion, enter the line number only.

Caution Be careful with names in a program. They must not include key words.

Token 134 **Name** Dimension

Class Array and String Command

DIM

Form Dim variable (subscript 1, subscript 2 . . .)

Conditions Reserves space in memory for an array of dimension equal to the number of subscripts.

Only one array of a given name can exist, even though dimensions are specified to be different. Only last entry is retained.

The number of dimensions is limited to 88.

The magnitude of the subscripts is limited only by available memory.

Arrays may be dimensioned by variables or by expressions, but variables must have been defined before DIM is encountered.

The number of elements in the array is equal to the product, (1 + subscript 1) × (1 + subscript 2) . . .

If not specifically set by DIM, an array is assumed to have 10 as the subscript value (11 elements, including the zeroeth).

Individual strings in a string array are not dimensioned, but grow or shrink as necessary. The maximum length is 255 characters.

Array element values are set to 0 or null by RUN or CLEAR. Use GOTO to avoid this.

Memory use is (per element)

- —integer numbers: 2 bytes
- —real numbers: 5 bytes
- —strings: 3 bytes plus 1 per character.

Token 148

Class Graphics and Games Command

DRAW

Forms DRAW Shape number AT X coordinate, Y coordinate

DRAW Shape number

Conditions In high resolution graphics, places a previously defined Shape identified by the Shape number (range 0-255) at the screen location X, Y.

Color, rotation, and scale of the Shape are defined by a table, set up by SHLOAD or by the monitor program. The number may be an expression.

If AT and coordinates are omitted, the Shape is placed at the last plot point defined by HPLOT, DRAW, or XDRAW.

Coordinates may be defined by expressions or variables or given explicitly. Maximum range of X is 0 to 278 and of Y is 0 to 191, both inclusive.

Use XDRAW to erase the Shape when desired.

Note Endless loops and random Shapes may occur if DRAW is used before a Shape table is entered.

See AT, HPLOT, XDRAW.

Token 128

Class System and Utility Command

END

Conditions Used to stop execution of a program and return control to keyboard user.

No message is printed.

After END, CONT causes execution resumption, starting at the next instruction, not the next line number. CONT has no effect if there is no further program.

Token **Name** Exponential

Class Mathematical Function

EXP

Forms EXP (number)

EXP (expression)

Conditions Returns the value of the base of natural logarithms (e) raised to the power of the number or evaluated expression.

e is evaluated to six places, e = 2.718289.

Token 159

Class Editing and Format Command

FLASH

Conditions Sets the screen display to alternate between normal and inverse characters (alternately white on black background, then black on white).

Continues to function until NORMAL (white on black) or INVERSE (black on white) is encountered.

FLASH affects output but not input.

Token 194 **Name** Function

Class Mathematical Function

FN

Form FN name (argument)

Conditions Returns the value resulting from the application of the previously named function (defined by DEF FN . . .) to the argument which replaces the dummy argument used in the definition.

The name may not be an integer type of variable, and string functions are not allowed.

The argument may be a number or an expression, and need not include the variable used in the definition.

FN can be used in immediate execution although the function can only be defined in the program execution mode.

See DEF.

Token 129

Class Flow of Control Instruction

FOR

Form FOR variable = initial value TO limit value STEP increment

Conditions The initial element of a FOR-NEXT loop.

The variable must be real, not an integer. Initial, limit, and increment values may be real or integer values.

The value of the variable is set to its initial value (a numerical or another variable or evaluated expression), and then execution continues until NEXT is encountered; the variable value is then incremented by the increment, if given, or by 1, if not. The result is tested against limit. If the limit is exceeded, the statement following the NEXT is executed; otherwise execution returns to the statement following the FOR construct.

The variable cannot be a string or integer variable.

Initial, limit, and increment values are not restricted to integer values.

A FOR-NEXT loop may be completely within another one (nested), but loops may not cross. A maximum of ten loops in a nest are allowed.

A FOR-NEXT loop can be run in the immediate mode if all parts, including any intermediate statements, can be placed on one line (maximum 239 characters).

See TO, NEXT, STEP.

Token 214 **Name** Free Memory

Class Editing and Format Function

FRE

Forms FRE

FRE (any legal expression)

Conditions FRE returns the amount of memory (in BYTES) still available to the user. If any negative number is returned, add 65536 to get the actual amount.

FRE (expression) returns the free space below string storage space and above the numeric array and string storage space. The expression is a dummy value, but must be evaluatable.

FRE (exp) also clears unused variable data from the area. LET X = FRE (0) can be included in a complex program to do this.

To check available memory during program entry, use PRINT FRE(0).

Code 190

Class Input-Output Command

GET

Form GET variable name

Conditions Fetches a single character from the keyboard (with some complexities). Does not display this, or require RETURN. Stores the value in the name variable.

Complexities:

- —Control @ returns a null character.
- —A← or Control H may cause a blank space on the screen.
- —Control C does not halt program execution.
- —For arithmetic variables, a colon, comma, plus, minus, Control@, Control E, space and period return a 0. A RETURN or nonnumeric character causes an error message and execution halt.

Hint To avoid most complexities, GET a string variable, then convert it to a numeric using the VAL function.

Token 176 **Name** GO to Subroutine

Class Flow of Control Command

GOSUB

Form GOSUB line number

Conditions Causes a program branch to a subroutine located at specified line number. Execution proceeds with ensuing lines, until RETURN is encountered. Program execution then is transferred to the statement following the GOSUB.

Subroutines may call other subroutines, up to a limit of twenty-four levels.

See RETURN.

Hint The search for the called line starts at line 0. Program execution will be faster if subroutines are placed early in the program with the ones called most often first.

Token 171

Class Flow of Control Command

GOTO

Forms GOTO line number

IF condition GOTO line number

Conditions Causes a jump to the specified program line, with program execution resuming with the line specified.

See also IF.

Token 136 **Name** Low Resolution Graphics

Class Graphics and Game Command

GR

Conditions Sets the low resolution graphics mode of forty lines of forty elements each, with space for four elements each, with space for four lines of text at the bottom of screen.

Clears screen to black, sets COLOR to 0, and cursor to the bottom text line.

The coordinate origin is the upper left, with X and Y varying from 0 to 39.

Low resolution graphics and text are stored in the same memory area. GR deletes the upper twenty lines of text.

HOME in GR mode sets cursor to bottom of screen.

Hint The text area can be converted to graphics by following GR with POKE −16302,0. The color pattern can be controlled by key usage.

In TEXT mode, POKE −16304,0 converts entire screen to graphics without clearing it. Colors may be controlled by key usage.

Token 146 **Name** High Resolution Color

Class Graphics and Games Command

HCOLOR

Conditions Sets high resolution graphics to color values designated by the integers 0-7 inclusive.

The exact color depends on the TV and its adjustment; for H COLOR = 5, white will be obtained only if both dots (X,Y) and (X+1,Y) are plotted. For X-odd, the color is green(ish) and for X-even, the color is blue(ish).

Until the first HCOLOR is executed, screen color is indeterminate.

HCOLOR is not changed by HGR, HGR2, or RUN. It has no influence on low resolution graphics.

See Codes and Designators for COLOR, HCOLOR.

Token 145 **Name** High Resolution Graphics,

Class Graphics and Games Command

HGR

Conditions Sets the high resolution graphics mode, 280 positions horizontally by 160 positions vertically. Four test lines may be displayed at the bottom of the screen.

The origin of X-Y coordinates is 0,0 at upper left of screen.

The screen is cleared to black, and Page 1 of the graphics memory is displayed.

While only four text lines are displayed, all lines are available for text (stored separately), and will be displayed when the TEXT command is used.

The cursor can be moved off screen by Escape I, or by HOME.

HGR is not available in Cassette Applesoft.

Notes POKE 49234,0 converts screen to 280 positions horizontally by 192 positions vertically (full screen).

In very long programs, high resolution screen and program data may overlap. HGR2 can sometimes be used to minimize the risk of this.

HGR2

Conditions Sets full screen, high resolution graphics, 280 positions horizontally by 160 positions vertically, and clears screen to black.

Blanks between the 2 and R are optional.

This is Page 2 of graphics memory, but it is not available with less than 24K of memory.

While text is not displayed, it is entered into text storage and is available by using the TEXT command.

Use HGR2 instead of HGR to maximize program memory. On 24K systems, setting HIMEM:16384 will protect the screen area from programs.

Token 163 **Name** SET High Memory Limit

Class System and Utility Command

HIMEM:

Forms HIMEM: number

HIMEM: variable

HIMEM: expression

Conditions Sets the upper bound of memory available to a BASIC program. The expression must be in the range −65535 to 65535 inclusive. However, practical maximums depend on installed memory, as follows:

16K	16384
32K	32767
64K	49152

HIMEM is a system variable, normally automatically set to highest available memory. PRINT PEEK (116)* 256 + PEEK (115) will return the current value.

Use to protect a memory area, say for a machine language program.

See LOMEM:.

Token 142 **Name** Place Horizontal Line

Class Graphics and Games Instruction

HLIN

Form HLIN start, end, AT line number

Conditions In low resolution graphics, HLIN places a horizontal line on the numbered line, starting and ending as designated.

Line values may be numerical, a variable or an expression, allowed ranges being:

Start-Line	0-39
End-Line	0-39
Line Number	0-47

Line color is set by the most recent COLOR statement.

In text, with COLOR set to other than zero, HLIN places a specified row of characters on screen. COLOR = 0 deletes any character in the specified row.

HLIN has no visible effect in the high resolution graphics mode.

Token 151

Class Editing and Format Command

HOME

Conditions In the TEXT mode HOME moves the cursor to the upper left corner and clears all text.

In GR and HGR modes, HOME moves the cursor to the upper left of text area and clears all text.

In HGR2, there is no visible effect.

Note HOME is identical to CALL –936 and to Escape @ return.

Token 147 **Name** High Resolution Plot

Class Graphics and Games Instruction

HPLOT

Forms
HPLOT X, Y
HPLOT TO X, Y
HPLOT X_1, Y_1 TO X_2, Y_2 TO X_3, Y_3 . . . .

Conditions Used to plot or draw in the high resolution graphics mode.

In simple form, HPLOT places an element at the X, Y coordinates; X can range from 0 to 279, Y can range from 0 to 159 (normal) or 0 to 191 (full screen), all inclusive.
If HPLOT is followed by TO, it draws a line from the last point plotted to the indicated coordinates.

In full form, it plots from coordinates X_1, Y_1 to X_2, Y_2 to X_3, Y_3 . . . . (up to the instruction limit of 239 characters).

The color is the last color specified by HCOLOR.

Warnings HPLOT must be preceded by HGR or HGR2 to avoid overwriting the program and its data.
An attempt to plot outside the limits (x from 0 to 279, Y from 0 to 191) produces an error.

Token 150 **Name** Horizontal Tabular Set

Class Editing and Format Instruction

HTAB

Forms HTAB number

HTAB variable

HTAB expression

Conditions Used to set the print position to the indicated value, which must be in the range 0 to 255.

Position 1 is the left most position of the line containing the cursor. Position 40 is the right most; position 41 is the left most position of the next line down and so on. TAB0 is evaluated as position 256, that is, position 16 of line 6 down.

Note HTAB cannot be used as an item in a print list. Precede and follow it with a semicolon to control the print position, which may precede an item already printed on the same line.

Warning It is possible to place one character off screen.

Token 173

Class Flow of Control Instruction

IF

Forms IF condition THEN Instruction

IF condition THEN GOTO line number

IF condition GOTO line number

Conditions IF sets up a branching test in the forms shown.

If the condition evaluates as nonzero, or true, the instruction following the THEN is executed.

If the condition evaluates as zero, or false, program execution is transferred to the next numbered line.

The general form THEN GOTO line number can be simplified by omitting either THEN or GOTO.

Mixed arithmetic-string expressions in conditions are evaluated by comparing the ASCII numerical values of characters.

A null string literal is evaluated as zero.

All other strings are evaluated as true.

Repeated string evaluation may halt the program.

Caution All instructions following THEN that are on the same line will not be executed if the condition evaluates as false. Multiple instructions on lines with IF-THEN constructs can be sources of errors.

Token 139 **Name** Select Input Slot

Class Input-Output Instruction

IN#

Form IN# slot-number.

Conditions Selects the peripheral or device plugged into a master board slot for subsequent INPUT.

The value of the slot may be a numeric value, a variable, or an expression.

The slots and devices are numbered 1-7 inclusive, and the values must be within this range.

IN#0 indicates that input will be from the keyboard.

The instruction boots the disk if the disk controller is present at the specified slot.

Warnings Values between 8 and 255 may create a hang condition. Values greater than 256 will produce an error.

Token 132

Class INPUT-OUTPUT

INPUT

Form INPUT "String"; variable 1, variable 2 . .

Conditions Used to request an input (normally identified on the screen by a string in quotes), and to accept values of the variable or variables listed.

The string to be printed on the screen must be separated from the variables by a semicolon, and the variables by commas.

Responses to a variable request must be appropriate to the variable type. The input values for the variables must be separated by commas. A leading comma or semicolon is evaluated as a null input.

Extra values are ignored; a message is printed; and execution continues.

The string and the semicolon may be omitted. In this case, only the ? prompt is printed.

If insufficient variables are input, a ? prompt results.

Token 211 **Name** Integer Value

Class Mathematical Function

INT

Forms INT (number)

INT (variable)

INT (expression)

Conditions Returns the largest integer that is less than or equal to the evaluated expression, current value of variable, or supplied number.

INT has no meaning for string quantities.

Token 158

Class Editing and Format Command

INVERSE

Conditions Used to set video output display to show black characters on a white background.

Does not affect display of input characters.

Use NORMAL to return to white on black display.

Use FLASH to alternate between the two.

Token 232 **Name** Left String Slice

Class Array and String Function

LEFT$

Form LEFT$ (object string, character select)

Conditions Applied to an object string, returns the n left-most characters.

The object string may be a literal, a string variable, or a string expression.

The character select may be a number, a variable, or an expression. A real number is converted to an integer (Range 1 to 255 inclusive).

See RIGHT$, MID$; use + for string concatenation.

Token 227 **Name** String Length

Class Array and String Function

LEN

Form LEN (string expression)

Conditions When applied to a string variable or expression, LEN returns the length of the string in number of characters, including spaces.

An error results if length of the string is greater than 255 characters.

Token 170

Class Input/Output Instruction

LET

Form LET variable = Value

Conditions Used to assign a value to a variable.

Either string or numerical assignments can be made, but the variable and value must be the same type. However a literal will be evaluated as numerical, if possible, if assigned to an arithmetic variable.

The variable may be superscripted.

LET also defines the variable if it has not been previously defined.

LET is optional. The form A = 2 is valid, and defines the variable if it has not been defined by a previous statement.

For numerics, the value may be a literal, another variable, or an expression; and for strings, the value may be a string, another string variable, or a string expression.

Token 188

Class Editing and Format Command

LIST

Forms LIST

LIST line number

LIST line number 1, line number 2

LIST line number

Conditions Displays the program in memory on screen, as follows

—if line number, that line only.

—if two line numbers separated by a comma (or other delimiter), from line number 1 to line number 2 inclusive.

—if a line number followed by a comma (or other delimiter), from line number to end of program.

Note (Some other forms are possible.)

Use SPEED = h to set rate of character display.

Reset terminates listing immediately. Control C terminates listing at the end of

the line being listed and transfers control to the keyboard. CONT resumes LIST.

Use Control S to suspend LIST. The pause occurs at the end of the line. Any key restarts it.

Token 182

Class System and Utility Command

LOAD

Conditions Used to input a program from magnetic tape to memory.

Tape operation is manual; press return before starting the tape.

When the data flow starts, any program in memory is erased.

Tape recorder volume level is critical.

Use RESET to interrupt the loading process.

Inclusion of LOAD in any name may cause loss of program and system hang-up.

Using LOAD with no tape recorder input available will cause the computer to wait indefinitely for input. Use RESET to escape. The current program is not lost.

In Disk Basic, LOAD becomes LOAD name plus optional parameters. This command will transfer the named program from disk to memory.

Token 220 **Name** Natural Logarithm

Class Mathematical Function

LOG

Form LOG (variable)

Conditions Returns the natural logarithm of the variable, which may be a number, a variable, or an expression (whose current value is used).

LOG has no meaning for string quantities.

Token 164

Class System and Utility

LOMEM

Forms LOMEM: number

LOMEM:

LOMEM: variable

LOMEM: expression

Conditions Sets the address of the lowest memory available to a BASIC program.

The value may range between −65535 and +65535. However, an out-of-memory error will occur if

- —Low memory is set higher than the High memory limit.
- —Low memory is set below the highest memory of the operating system plus any stored program (about location 2051).

To see current value, use PRINT PEEK (106) + 256 * PEEK (105).

Hint On 24K systems, set Low Memory to 16384 to protect variables from high resolution graphics.

LOMEM: is reset by NEW or DEL, or by adding or changing a program line, and by Control B, which deletes the program.

Warning Changing LOMEM during program execution can cause execution errors.

Token 234 **Name** Middle String Slice

Class Array and String Function

MID$

Forms MID$ (object string, left character select, number character select)

MID$ (object string, left character select)

Conditions Applied to an object string, MID$ returns the characters starting with the one in the position indicated by the left character select value, and proceeding to the right for the number of characters designated. If the number character select value is omitted, selection proceeds to the end of the string.

The object string may be a literal, or may be designated by a string variable or string expression.

The character selectors may be numbers, variables, or expressions. A real number is converted to an integer (range 1-255 inclusive).

See LEFT$, RIGHT$: Use + for string concatenation.

Token 191

Class System and Utility Command

NEW

Conditions Used to prepare for a new program.

Deletes the current program and all variable assignments.

Does not actually clear the program and variable values in memory. It simply removes the pointers.

Note Special programs that can recover stored variable values exist.

Token 130

Class Flow of Control Instruction

NEXT

Forms NEXT variable name

NEXT variable name, variable name, . .

NEXT

Conditions The second, or terminating, element of a FOR-NEXT loop.

The action occurring when NEXT is encountered depends on the test set up in the FOR statement. If test conditions are not satisfied, the sequel following the FOR statement is executed. If the test is satisfied, execution continues with the statement following the NEXT.

If several loops end at the same point their identifying variables may follow the NEXT. The sequence must be correct.

If the variable name is omitted, the NEXT is assumed to apply to the most recent FOR.

In immediate execution, the FOR and NEXT, and all intervening statements, must be entered before RETURN is pressed.

See FOR.

Hint FOR-NEXT loops execute faster if the variable name is omitted in the NEXT statement.

Token 157

Class Editing and Format Command

NORMAL

Conditions Sets the display mode to white letters on a black background for both input and output.

See INVERSE, FLASH.

Token 198

Class Relational Operator

NOT

Form NOT variate

Conditions The relational inverting operator NOT converts a logical true value (1) to a logical false value (0), and vice versa.

NOT acts as a unary operation, affecting the immediately following variate. Parenthesis must be used to apply it to an entire expression.

NOT has no meaning for string variables.

Examples NOT 0 = 1

NOT 1 = 0

NOT 9 = 0

NOT −9 = 0

Token 156

Class System and Utility

NOTRACE

Conditions Terminates a debug mode set up by TRACE, which is used to print line numbers on the screen tracing program execution.

See TRACE.

Token 180

Class Flow of Control Instruction

ON

Forms ON condition GOTO line number 1, line number 2 . . .

ON condition GOSUB line number 1, line number 2 . . .

Conditions If the condition evaluates to 1, ON causes a branch to the first line number listed. If it evaluates to 2, a branch is made to the second line number listed, and so on.

If the condition evaluates to 0 or to a number greater than the number of line numbers listed, execution proceeds to the next program statement.

The condition may be a variable or an expression, but it must evaluate to less than 256 but not less than 0.

Token 165 **Name** On Error Routine

Class Flow of Control Instruction

ONERR

Form ONERR GOTO line number

Conditions Use to avoid having an error message printed and execution halted when an error occurs in program execution. (Typical use: to detect the divide by zero encountered error.)

Must be executed at least once before error is encountered to prevent execution halt with error message.

Use RESUME at end of error avoidance routine (but see below). Execution resumes at beginning of error producing statement.

Warnings Within FOR-NEXT loops, or between GOSUB and RETURN, return should be to the FOR or GOSUB, not to the error producing statement. The service routine must do the necessary cleanup.

Token 206

Class Relational Operator

OR

Form Relation 1 OR relation 2

Conditions The arithmetic logical operator OR assumes the value true if either relation 1 or relation 2 is true, or if both relations are true.

There may be more than one OR in sequence. OR has the lowest priority in expression evaluation.

May be combined with NOT in the forms NOT relation 1 OR NOT relation 2, and so forth.

OR has no meaning for string expressions.

Relations are true if nonzero

Truth table for OR

X	Y	X OR Y
F	F	F
F	T	T
T	F	T
T	T	T

Code 216 **Name** Read Paddle Input

Class Graphics and Games Function

PDL

Form PDL (paddle number)

Conditions Reads the current paddle position (as an integer between 0 and 255) of one of four paddles specified by a paddle number, which may be a number, a variable, or an expression in the range 0-3.

Note Paddle input involves an analog to digital conversion, which is slower than machine operation. Allow several program lines between two successive paddle reads by using a program loop if necessary.

Warning Attempts to select a paddle number greater than three may affect program execution.

Hint The paddle control circuit is essentially a 0-150K variable resistor. It is useful in many control and measurement applications.

Token 226

Class System and Utility Function

PEEK

Form PEEK (address)

Conditions Returns the byte stored at the given address. Both address and the returned quantity are in decimal numbers. The address may be given as a number, a variable, or an expression, and may be positive or negative.

See POKE.

Note Thorough understanding of system variables and their addresses is necessary to obtain maximum use of PEEK and its companion, POKE.

Minus address = true address – 65535.

Token 141

Class Graphics and Games Instruction

PLOT

Form PLOT X coordinate, Y coordinate

Conditions In low resolution graphics, places a dot, a pixel or picture element, at the position given by the coordinates.

The coordinate value may be given numerically, by a variable, or by an expression. X values must be in the range 0-39, and Y values, in the range 0-47, with 0,0 at the upper left.

The color plotted is that given by the most recent COLOR statement; or COLOR = 0 is used if COLOR has not been previously specified.

In the Text or Graphics plus Text modes, for Y in range 40-47, PLOT places a character at the location specified. The character code corresponds to the color code.

There is no visible effect in the High Resolution Graphics mode.

Token 185

Class System and Utility Command

POKE

Form POKE address, quantity

Conditions Stores the (byte) quantity at the address indicated. Both quantity and address are in decimal and either may be specified by a number, a variable, or an expression. The maximum range of addresses is −65535 to 65535, and of the quantities 0 through 255. Real values are converted to integers.

Receiving hardware must be present at the address for successful use. For Memory, this is 0 to

16K - 16384
32K - 32768
48K - 49152

Memory mapped peripherals are normally at $C000 through $C07F (decimal 49152 through 49279) (or −16384 through −16257).

See PEEK.

Note Thorough understanding of system variables and their addresses is necessary to obtain maximum use of POKE and its companion PEEK.

Warning Careless use of POKE may alter system and/or program performance.

Token 161 **Name** POP RETURN STACK

Class Flow of Control Command

POP

Conditions "Pops" or removes one address from the GOSUB-RETURN address stack.

The effect is to return to the second-most recent GOSUB, rather than to the most recent.

Produces an error signal if there are no addresses on the stack.

Token 217 **Name** Read Cursor Position

Class Editing and Format Function

POS

Form POS (dummy expression)

Conditions Returns the column number or horizontal position of the screen cursor.

The expression is a dummy but must be evaluatable as a numeric.

Note For POS and SPC, positions are numbered from 0, while for HTAB and TAB they are numbered from 1.

Hint POS (8) or POS (9) are simple forms.

Token 186

Class Input-Output Instruction

PRINT

Form PRINT print-list

Conditions Causes the items of the print-list to be displayed on the screen.

An empty list causes a return and line feed.

A single item in the print list, or a last item with no following punctuation, is printed on its own line.

Two or more items separated by semicolons are printed with no intervening spaces. A terminal semicolon holds the print position until the next print list.

If separated by a comma, the next item is placed in the next available tabular location.

Items in quotes (strings) are printed without the quotes.

In a print list of numbers followed by

periods, usually interpreted as a literal, the terminal period prints as 0.

The maximum total string length is 255 characters.

Numerical values may use from 1 to 19 print positions.

See TAB, SPC.

Hints The range of variation in the print-list is very large. Practice trials are recommended.

A ? can be used in lieu of PRINT. It lists as PRINT.

Token 198 **Name** Select Output Slot

Class Input-Output Command

PR#

Form PR# slot-number

Conditions Transfers output to the peripheral located in the designated slot.

The slot number may be a numeric, a variable, or an expression, and must evaluate to the range 1-7 inclusive.

A value of 0 returns output to the TV screen.

A value less than 0 or more than 7 produces an error report.

PR# boots the disk if the specified slot contains a disk controller.

Warnings If there is no peripheral in the specified slot, the system will hang.

Use Control C and RETURN to escape.

Token 135

Class Input-Output

READ

Form READ variable 1, variable 2, ...

Conditions When encountered, READ sets the value of the specified variable or variables to the value of the elements of DATA lists, starting from first variable in the first READ statement and the first value in the DATA statement, and continuing through the DATA list or lists until the READ list or lists are satisfied.

If the DATA lists are shorter than READ lists, an error message results. However, there is no indication if the DATA list is longer.

In immediate mode, READ attempts to find DATA lists in the stored program.

The position in the READ list is recorded. It may be set to the beginning of the DATA list by the RESTORE statement. (It is not automatically reset by an immediate mode execution of a READ.)

Token 167 **Name** Recall Tape Stored Array

Class Array and String Command

RECALL

Form RECALL name

Conditions Recalls an array stored on tape and reads its values into the named array.

Since STORE does not actually store the name of its named array, any input array will be accepted.

Unless the requested and stored arrays match in dimensions, scrambled numbers, false zeros, and/or error messages will be encountered.

Arrays must be numeric. (String arrays may be converted using the ASC statement.)

Tape operation is manual. An audible signal occurs at the beginning and end of the tape record.

The RECALL function can be interrupted only by RESET.

See STORE.

Warning While useful in systems having no disk drive, care is necessary. Review the instruction book.

Token 178 **Name** Remark

Class Editing and Formatting Command

REM

Form REM characters

Conditions REM is a command to ignore, during program execution, all following characters (including statement separators and blanks), until a return (line feed) is encountered.

There is no limit to REM length, (up to available memory).

Token 174 **Name** Restore Data List Pointer

Class Input-Output Command

RESTORE

Conditions Resets the data list pointer to the beginning of the data list.

See READ, DATA.

Note The beginning of the data list is the first value following the first DATA statement in the program.

Token 166 **Name** Resume Execution

Class Flow of Control Command

RESUME

Conditions Used at end of an error handling routine to cause program execution to resume at the beginning of the statement that contained an error.

Warnings If the error handling routine has not correctly dealt with the error, an endless loop results. Use Control C, and RETURN to escape.

If RESUME is encountered before an error has occurred, the effect may be unpredictable. Usually execution will cease.

In the immediate mode, RESUME may have an unpredictable effect, including initiation of existing or deleted programs.

Token 177

Class Flow of Control Command

RETURN

Conditions The signal to end a GOSUB subroutine.

When encountered, RETURN causes a branch to the statement following the most recently executed GOSUB; that is, the address of the statement at the top of the GOSUB stack.

Using a RETURN without a GOSUB or having more RETURNs than GOSUBs causes an error message.

Note POP affects the GOSUB order of execution, but continue to watch the number of RETURNs.

See GOSUB, POP.

Token 233 **Name** Right String Slice

Class Array and String Function

RIGHT$

Form RIGHT$ (object string, character select)

Conditions Applied to the object string, this function returns the number of right-most characters indicated by the character select value.

The object string may be a literal, a string variable, or an expression.

The character select may be a literal, a variable, or an expression. A real number is converted to an integer (range 1 to 255 inclusive).

See LEFT$, MID$. Use + for string concatenation.

Token 219 **Name** Random Number

Class Mathematical Function

RND

Form RND (indicator)

Conditions RND returns a pseudorandom number; that is, one of a distribution that is random by test but that is generated according to a rule.

The indicator may be a number, a variable, or an expression.

If the indicator is a negative value, a particular random number that is always the same for that negative number is generated. Subsequent positive arguments return a particular, repeatable sequence.

If the indicator is positive, without a prior negative value, a new random number is generated for each use.

If the indicator is 0, the most recent random number is returned.

The value of RND ranges from 0 to 1, but is never unity.

Token 152 **Name** Rotate Shape

Class Graphics and Games Instruction

ROT=

Form ROT = Angle

Conditions ROT= causes a previously specified shape to be rotated clockwise about its reference point; that is, it sets the angular rotation of the shape to be placed on the screen by DRAW or XDRAW.

The angle is based on 64 parts of a circle, that is, angle = 32 is a rotation of 180° clockwise.

The value of angle is affected by SCALE. For SCALE = 1 only the four values 0, 16, 32, 48 are recognized. For SCALE = 2, eight values are recognized, and so on.

Intermediate values usually give the next smaller recognized value.

The angle may be specified by a number, a variable, or an expression. Real values are converted to integers, which must be in the range 0-255.

Token 172

Class System and Utility Command

RUN

Forms RUN

RUN line number

Conditions Clears all variables, internal pointers, and stacks, and commences program execution at the indicated line number, if present, or at the next higher one. If no number is given, execution starts at the lowest line number encountered.

To execute a program without clearing variables, use GOTO or GOSUB.

RETURN executes RUN immediately.

Control C terminates RUN at end of current statement and transfers control to the keyboard. CONT resumes program execution.

Control S suspends RUN at end of any statement. Any key causes it to continue.

Note In DISK BASIC RUN becomes RUN name plus optional parameters.

Token 183

Class System and Utility Command

SAVE

Conditions SAVE initiates serial output to a jack, to be fed to a magnetic tape recorder to allow the saving of the program for later reentry using the LOAD command.

Recorder operation is manual. The recorder must be connected, volume level set, and the tape running before the command is executed, to avoid recording errors.

The beginning and end of data flow is signaled by an audible beep.

Hint Do not use a recorder with automatic level control. Level adjust is critical. (Record the setting that gives good results, or use a level indicator.)

Note In DISK BASIC, SAVE name plus optional parameters is used. BSAVE name, A-address, L-length plus optional parameters is the related command to transfer memory to disk.

Token 153

Class Graphics and Games Command

SCALE=

Form SCALE = multiplier

Conditions Sets the scale for a shape to be drawn by DRAW or XDRAW, by multiplying each vector in the shape table by the indicated multiplier.

The multiplier may be a number, a variable, or an expression. Real values are converted to integers and must be in the range 0-255 inclusive.

A multiplier of 1 gives a 1:1 scale shape reproduction, and of 2 gives a 2:1 scale, and so on up to 255:1 scale. However, 0 gives the largest magnification, 256.

Token 215 **Name** Screen Color

Class Graphics and Games Function

SCRN(

Form SCRN (X coordinate, Y coordinate)

Conditions In low resolution graphics, SCRN(returns the screen color code of the point defined by the coordinates.

The coordinates may be expressed by a number, a variable, or an expression. Real values are converted to integers. The evaluated expression must be in the range 0 to 39 inclusive for the X coordinate, and 0 to 47 inclusive for the Y coordinate to give a true return. Values up to 47 will be accepted for the X coordinate, but they will return values related to text or to items not on the screen.

In high resolution graphics, SCRN(returns the color (or character) last recorded in the low resolution graphics memory area.

In the text mode, SCRN(returns half of the character code, for the character at (X + 1), INT$\frac{(Y + 1)}{2}$, the upper half if

the Y coordinate is odd and the lower half if it is even. The expression:

```
CHR$(SCRN(X-1,2*(Y-1) +
  16*SCRN(X-1),
```

2*(Y-1) + 1) returns the character code at character position (X,Y).

Token 210 **Name** Signum Function

Class Mathematical Function

SGN

Form SGN(variate)

Conditions The signum function SGN returns the value of −1 if the variate is negative, +1 if it is positive, and 0 if it is zero.

The variate may be a number, an unknown, or an expression.

Signum has no meaning for string values.

Token 154 **Name** Load Shape Table

Class Graphics and Games Command

SHLOAD

Conditions Loads a shape table from magnetic tape into memory just below HIMEM, which is then moved to protect the shape.

Warning In 16K or 32K systems, using SHLOAD may result in program or graphics loss.

Hint If a shape in memory is to be replaced, reset boundary by entering HIMEM: 8192 before loading the new shape. This wipes out the old shape and saves memory.

See Shape Table.

Token 223 **Name** SINE Function

Class Mathematical Function

SIN

Form SIN(angle)

Conditions Returns the sine of an angle expressed in radians.

The angle may be given by a number, a variable, or an expression.

SIN has no meaning for string values.

Token 195 **Name** Skip Spaces

Class Editing and Format Instruction

SPC(

Form ;SPC(spaces);

Conditions A print-list instruction that introduces the indicated number of spaces between the last print item (or left screen) and the next item.

Spaces may be designated by a number, a variable, or an expression. Real values are converted to integers. The number of spaces must be between 0 and 255 inclusive.

The instruction may be repeated.

Use of punctuation follows print-list rules.

SPC must be in a print list.

See PRINT.

Token 169

Class Editing and Format Command

SPEED=

Form SPEED = rate

Conditions Sets the rate at which characters are sent to the screen or to another output device.

The rate may be a number, a variable, or an expression. Real values are converted to integers, which must be in the range 0-255 inclusive.

0 corresponds to about two characters per second, 255 gives about 50 per second.

Token 218 **Name** Square Root

Class Mathematical Function

SQR

Form SQR(variate)

Conditions Returns the positive square root of the variate.

The variate may be a number, a variable, or an expression.

Negative values produce an error.

SQR has no meaning for string variables.

SQR executes faster than the equivalent X ∧ 0.5.

Token 199

Class Flow of Control Instruction

STEP

Form FOR variate = initial value TO limit STEP increment

Conditions Sets the value of the increment for each passage through a FOR-NEXT loop.

The increment may be a number, a variable, or an expression.

The increments do not need to be integral values, but may be integers, as may the initial and limit values. The variate must be real.

See FOR, TO, NEXT.

Token 179

Class System and Utility Command

STOP

Conditions When encountered, STOP suspends execution of a program at the end of the statement being processed. A check for STOP is made at the end of each statement during processing. Control returns to keyboard. Nothing is cleared. The line number containing the executed STOP is printed.

Control C has the same effect, but clears some pointers and stacks.

CONT causes execution to resume with the next statement.

See CONT.

Token 168

Class Array and String Command

STORE

Form STORE array-name

Conditions Used to record the values of the components of an array on magnetic tape.

The array-name designates the source of the data. It is not recorded on tape.

Only real and integer arrays can be stored. String arrays must be converted using the ASC function.

Tape operation is manual. The recorder must be properly set and running before the command is executed to avoid error. A beep signals the start and end of actual transmission.

STORE can be interrupted only by reset.

See RECALL.

Warning While useful in systems having no disk, care is necessary. See the instruction book.

Token 228 **Name** String Convert Function

Class String and Array Function

STR$

Form STR$(value)

Conditions Converts a value into a string.

The value may be a number, an unknown, or an expression, and may be real or integer.

STR$ has no meaning for string quantities.

Token 192 **Name** Tabular Function

Class Editing and Format Command

TAB(

Form TAB (spaces)

An element of a print list, usable only in print lists.

Conditions Moves the print position to the indicated number of spaces from the left margin if the number of spaces is greater than the present position; otherwise it does nothing. (Use HTAB if left movement is needed.)

Spaces may be designated by a number, a variable, or an expression. Real values must be converted to integers, which must be in the range 0-255 inclusive. 0 gives 256 spaces.

Token 224 **Name** Tangent Function

Class Mathematical Function

TAN

Form TAN(Angle)

Conditions Returns the tangent of an angle expressed in radians.

The angle may be given as a number, a variable, or an expression.

TAN has no meaning for string variables.

Token 137

Class Graphics and Games Command

TEXT

Conditions In the graphics or high resolution graphics modes TEXT, returns operation to the full screen text mode of twenty-four lines. The prompt and cursor move to the last screen line.

In the text mode, TEXT moves the prompt and cursor, (it is equivalent to VTAB 24). If a partial screen is set, TEXT resets it to full screen.

Token 196

Class Flow of Control Instruction

THEN

Forms IF condition THEN instruction

IF condition THEN GO TO line number

IF condition THEN line number

Conditions Basically, the second part of an IF THEN test, establishing the instruction or line number to be executed if the condition is satisfied.

Either THEN or GOTO can be omitted in the compound statement form.

See IF, GOTO.

Token 193

Class Graphics and Games

TO

Forms HPLOT TO X, Y

HPLOT TO X_1, Y_1 TO X_2, Y_2 TO X_3, Y_3 . . .

Conditions As part of a HPLOT instruction, TO is used to terminate a line at the indicated X, Y coordinate. TO can be followed by one or more coordinates continuing the line from the last point plotted to the new X, Y coordinate, and so on.

See HPLOT.

As part of a FOR-NEXT loop, TO establishes the limit value of the loop.

See FOR, NEXT.

Token 155

Class System and Utility Command

TRACE

Conditions A debug aid, which displays the line number (preceded by #) on the screen each time a statement is executed. The number may be overwritten by PLOT Commands.

Continues until NO TRACE or Control B is given.

Hint TRACE is especially useful when a conditional branch does not appear to be working as expected.

Token 213 **Name** User Function

Class System and Utility Function

USR

Form USR (argument)

Conditions This function passes an argument to a machine language subroutine.

The argument may be a number, a variable, or an expression, which evaluated and the result placed in the floating point accumulator (address $9D through $A3).

Address $0A through $0C must contain a machine language Jump (JMP, Low Byte, High Byte of address) to the starting address of the routine.

The result of the routine is placed on the floating point accumulator at execution of the required RTS, which ends the routine.

Note Use the monitor (CALL-151) to load the routine(s) and calling item(s). This can be done using POKE commands also.

Token 229 **Name** String Value Function

Class Array and String

VAL

Form VAL (string value)

Conditions This function makes a numerical evaluation of a string. The first character must be a possible number item (Space, d.p., +, –, E or 0 - 9), or a 0 is returned. Subsequent possible number items are returned, until a nonnumber item is encountered. It and all subsequent characters are ignored.

The string value may be a string, a string variable, or a string expression. If it is a string, it must be enclosed in quotes.

Token 143 **Name** Draw Vertical Line

Class Graphics and Games

VLIN

Form VLIN Start-Y, End-Y AT X

Conditions In low resolution graphics, VLIN draws a vertical line from the line designated as Start-Y to the line designated as End-Y, and located at the column designated as X. The color is that set by the most recent COLOR statement; that is, it must follow COLOR= _____.

The positions may be given by numbers, variables, or expressions. Y values must be in the range 0 - 47 inclusive, and the X value in the range 0 - 39 inclusive.

In the TEXT mode, or with graphics plus text for lines 40 - 47 inclusive, the line becomes a line of characters.

There is no visible effect in high resolution graphics.

See HLIN, AT.

Token 162 **Name** Cursor Tabulate

Class Editing and Format Instruction

VTAB

Form VTAB line

Conditions Moves the cursor to the specified line. Positions are relative to the top of the screen, and must be in the range 1-24 inclusive. (Action is independent of the text window, so printing may not appear on the screen.)

The line may be specified by a number, a variable, or by an expression. Real values are converted to integers.

See HTAB.

Token 181

Class System and Utility Instruction

WAIT

Form WAIT address, Condition, Test-on

Conditions Used to insert a pause of controlled duration into a program.

The address is the decimal address of a location whose contents is to be tested.

The condition is a numerical value 0-255 inclusive, which specifies the bits to be tested.

The test-on value may be from 0 to 255 inclusive. Effectively, this indicates whether the test for each bit is to be made for 1s or 0s. If neither is specified, 0s are assumed.

The test is made by the AND of the location content and the Condition. If any one of the bits matches (1s or 0s), the AND result becomes nonzero, and the wait is over.

The parameters may be numbers, variables, or expressions, and are converted to integer values.

Token 149 **Name** Complement-draw

Class Graphics and Games

XDRAW

Form XDRAW shape number AT X-coordinate, Y-coordinate XDRAW shape number.

Conditions Identical in construction to DRAW, but draws the shape in the complement to the color specified by the most recent HCOLOR command.

It's primary purpose is to provide a way to erase a shape (DRAW 3 followed by XDRAW 3 draws then erases the Shape 3, leaving only the specified background).

See DRAW, AT.

Token Not given **Name** Not given

Class

XPLOT

Condition XPLOT is a reserved word not currently used.

Appendix A
Variables

There are three types of variables—integer, (range+/−32767), real, (range +/−9.99998999 E±37) and string (0 to 255 characters).

Variables are created by naming them in an INPUT, LET, or DIM statement (LET is optional), or by encounter in a program.

Variable names can be a single letter, two letters, or a letter followed by a number, for a total of 936 possible simple variables.

Assigned names can be longer than two characters, but only the first two characters are used by BASIC (APPLESOFT=APPLE=AP as a name). Remaining characters are ignored unless they include a reserved word, which produces an error signal and a halt. (SING and SINGE are examples.) Names may not include key words.

If the name is followed by %, it indicates an integer variable, and if by $, it indicates a string variable. X, X%, and X$ are different variables.

Variables used as counters with FOR must be real.

A variable name followed by a number or numbers

in parentheses (subscripts) indicates an array variable. The array dimension is equal to the number of subscripts, which must be separated by commas, (maximum of 88 dimensions). Each subscript can vary from 0 to the number assigned in the DIM statement. If DIM is not used, the maximum subscript value is assumed to be ten. (However, string arrays can have up to 255 characters per element.)

Array names are independent of simple names (A(6)≠A).

Two arrays of different dimensions but with the same name are not allowed.

Appendix B
Delimiters, Operators, and Priorities

Delimiters

~	^	,
	>	;
=	<	:
-	/	(
+	*	)

Arithmetic Operators

=	*
+	/
-	^

Logical Operators

AND	>	><	<=
OR	<	>=	=<
=	<>	=>	

Unary Operators

+
-
NOT

Expression Evaluation Priority

Highest	( )	
	+, −, NOT	(unary operators)
	∧	(exponentiation)
	*,/	
	+,−	(binary operators)
	> , < , >=, <=, =<, <> , >< ,=	
	AND	
	OR	

Note: In the above, commas are separators only.

Appendix C
Control Character Usage

Control B = Select integer or full BASIC, depending on firmware setting.

Control C = Stop execution. (Also transfers from monitor to full BASIC.)

Control D = Use in DOS to transfer control from a BASIC line to DOS, for LOAD name and so on. See the DOS Manual.

Control X = (immediate mode) Enter a back spash and ignore line. A null response to INPUT. Will interrupt LIST, RUN execution, or INPUT if it is the first character.

Escape S = Stop listing. If repeated, resume listing.

Escape I = Move cursor ↑

Escape J = Move cursor ←

Escape K = Move cursor →

Escape M = Move cursor ↓

Escape F = Clear from cursor to end of screen.

Escape @ = HOME (clear screen, cursor upper left)

Control U = →

Control H = ←

Control G = Bell

Control S = Stop Listing and Program execution. Any key resumes execution.

In monitor mode:

Control U = Space

Control H = Backspace

Control K = Line feed

Control M = Jump to memory read routine.

Appendix D

DOS Commands

APPEND f { }
BLOAD f, [Aa], {}
BRUN f [,Aa] {}
BSAVE f , Aa, Lj {}
CATALOG [,Ss] [,Dd]
CHAIN f { }
CLOSE [f] { }
DELETE f { }
EXEC f [,Rp] { }
FP { }
INIT f { }
INT

IN#s
LOAD f { }
LOCK f { }
MAXFILES n
MON [*c*] [*l*] [*e*]
NOMON [*c*] [*l*] [*e*]
OPEN f [,Lj] { }
PR# s
POSITION f ,Rp
READ f [,Rr] [,Bb]
RENAME f,g { }
RUN f { }
SAVE f { }
UNLOCK f { }
VERIFY f { }
WRITE f [,Rr] [,Bb]

Notation

f-file name
g-another file name
s-slot number 1-6
v-volume number of a diskette
d-drive no. 1-2
p-position no.
r-record no.
a-address in RAM
b-byte no.
j-length specifier
n-number of files to be active at one time
[ ] = optional
CAPS required indicator

{ } [,Ss] [,Dd] [,Vv]

-display commands
-display inputs
-display outputs

Appendix E
Color Codes and Designators

Codes and Designators for COLOR

Code	Designator	Name
0	BLAK	Black
1	MGTA	Magenta
2	DBLU	Dark Blue
3	PURP	Purple
4	DGRN	Dark Green
5	GREY	Grey
6	MBLU	Medium Blue
7	LBLU	Light Blue
8	BRWN	Brown
9	ORNG	Orange
10	GREY	Grey
11	PINK	Pink
12	LGRN	Green
13	YELO	Yellow
14	AQUA	Aqua
15	WITE	White

Codes and Designators for HCOLOR

0	black 1
1	green (depends on TV)
2	blue (depends on TV)
3	white 1
4	black 2
5	(depends on TV)
6	(depends on TV)
7	white 2

Appendix F
Memory Saving Hints

Delete all REM statements.
Use multiple statements per line (but watch entry jumps).
Use integer arrays if possible.
Use variables instead of constants.
The terminal END is not necessary.
Reuse variables. This is usually easy for counters.
Use the zero elements of arrays.
Use GOSUB for functions to be performed two or more times.
If variables are reassigned, the old variable can be cleared by FRE (0).

Appendix G
Useful Internal
Routines Available as Calls

CALL-151 CALL 65385: Places computer in monitor mode, indicated by *.

CALL-868 : Clears the current line from cursors to right margin. Same as Escape E.

CALL-922 : Issues a line feed. Same as Control J.

CALL-912 : Scrolls text up one line. The top line is lost and the bottom becomes blank (for lines in the text window).

CALL-1994 : Normally sets the upper 20 lines of text Page 1 to@. In Page 1 low resolution graphics, it clears the upper 40 lines to black. It has no effect on Page 2 or high resolution graphics.

CALL-1998 : Normally clears text Page 1 to reversed@. In Page 1 low resolution full page graphics, it clears screen to black. It has no effect on Page 2 or high resolution graphics.

CALL 62450: Clears current high resolution screen to black.

CALL 62454: Sets current high resolution screen to the HCOLOR most recently used by HPLOT.

CALL-936 CALL 64600 : Moves cursor to upper left screen within scrolling position and clears all text within window. It is identical to HOME and Escape @ return.

Appendix H
Shape Table

A shape table is a set of bytes that define vectors and plotting points. Each byte has the form:

mmpmmpmm

which is evaluated from right to left. The mm indicates plot position moves, one screen call at a time, as follows:

mm =	00	Move up
	01	Move right
	10	Move down
	11	Move left

and the p indicates the plot instruction, as follows:

p =	0	Don't plot
	1	Plot

subject to the rules

- —If all remaining bits are zero, ignore all remaining bits.
- —If all bits are zero, the shape definition is complete. Each byte can thus contain a maximum of two plot points, and one move (not up).
- —The starting point is the coordinate given by DRAW and XDRAW. The scale is set by SCALE, and the orientation of "up" by ROT.
- —Hint: Graph paper and eight column "financial" paper are a necessity for fast shape generation, as is practice.

Appendix I
6501-6505 Operation Codes

Instruction	Short Description	Addressing Mode												
		Immediate	Zero Page	Zero Page, X	Absolute	Absolute, X	Absolute, Y	(Indirect, X)	(Indirect), Y	Accumulator	Implied	Relative		
ADC	ADD A	69	65	75	6D	7D	79	61	71					
AND	AND A	29	25	35	2D	3D	39	21	31	0A				
ASL	Shift L 1		06	16	0E	1E						90		
BCC	Branch Carry											B0		
BCS	Branch Carry											F0		
BEQ	Branch 0													
BIT	Test Bit		24		2C							30		
BMI	Branch-											D0		
BNE	Branch ≠ 0											10		
BPL	Branch +										00			
BRK	Break											50		
BVC	Branch O'Flo											70		
BVS	Branch O'Flo										18			
CLC	Clear Carry										D8			
CLD	Clear Decimal										58			
CLI	Clear I Flag										B8			
CLV	Clear O Flag													
CMP	Compare A	C9	C5	D5	CD	DD	D9	C1	D1					
CPX	Compare X	E0	E4		EC									
CPY	Compare Y	C0	C4		CC									
DEC	Decrement		C6	D6	CE	DE								
DEX	Decrement X										CA			
DEY	Decrement Y										88			
EOR	XOR A	49	45	55	4D	5D	59	41	51					
INC	Increment		E6	F6	EE	FE								
INX	Increment X										E8			
INY	Increment Y										C8			
JMP	Jump				HC (6C)									
Number of Bytes		2	2	2	3	3	3	2	2	1	1	2		

( ) indirect

6501-6505 Operation Codes (Cont)

Instruction	Short Description	Addressing Mode: Immediate	Zero Page	Zero Page, X	Absolute	Absolute, X	Absolute, Y	(Indirect, X)	(Indirect), Y	Accumulator	Implied	Relative
JSR	Jump w. Return				20							
LDA	Load A	A9	A5	B5	AD	BD	B9	A1	B1			
LDX	Load X	A2	A6	*B6	AE		BE					
LDY	Load Y	A0	A4	B4	AC	BC						
LSR	Shift Right		46	56	4E	5E				4A		
NOP	No Op										EA	
ORA	OR A	09	05	15	0D	1D	19	01	11			
PHA	Push A										48	
PHP	Push Sta.										08	
PLA	Pull A										68	
PLP	Pull Sta.										28	
ROL	Rotate Left		26	36	2E	3E				2A		
ROR	Rotate Right		66	76	6E	7E				6A		
RTI	Ret. int.										40	
RTS	Ret. S.R.										60	
SBC	SUB. A	E9	E5	F5	ED	FD	F9	E1	F1			
SEC	Set Carry F.										38	
SED	Set Decimal										F8	
SEI	Sel Int. F.										78	
STA	STO A		85	95	8D	9D	99	81	91			
STX	STO X		86	*96	8E							
STY	STO Y		84	94	8C							
TAX	A→X										AA	
TAY	A→Y										A8	
TSX	SP→X										BA	
TXA	X→A										8A	
TXS	X→SP										9A	
TYA	Y→A										98	
Number of Bytes		2	2	2	3	3	3	2	2	1	1	2

( ) Indirect
* Zero Page,Y

Computer Companion for the Apple II®/*/le*®

by Robert P. Haviland

- Instant access to all the information you need to create programs and get them running smoothly!
- All the key words for Apple II/*/le* arranged in easy-to-use alphabetical order!
- Includes formats, conditions and results of use, and tokens used for internal storage!
- Each key word is listed on a separate page . . . with tab index for quick reference!
- Lie-flat comb binding for easy use . . . printed in large, easy-to-read type on heavy card stock!

Have all the information you need for programming your Apple right at your fingertips . . . when and where you need it! This exceptional sourcebook is designed for quick and easy reference . . . conveniently arranged to refresh your memory on available terms and the specifics of how those terms should be written.

Includes all the reserved words used in Applesoft BASIC, and gives you an outline of control characters, a listing of DOS commands, codes and designators for color use, hints for saving memory space, a listing of available internal routines, data on creating shapes, and the complete assembly language instruction set.

If you own or have access to an Apple II or *//e* . . . this is the most practical programming helpmate you can have!

Robert P. Haviland is a professional engineer with extensive experience in the design, construction, and operation of home computers. A well-known author, he has written several best selling computer books for TAB.

FPT > $10.25

ISBN 0-8306-1603-9

PRICES HIGHER IN CANADA

995-1283